Model-Based Systems Engineering Meets DevSecOps

Featuring Jerome Hugues and Joe Yankel as Interviewed by Suzanne Miller

----

Welcome to the SEI Podcast Series, a production of the Carnegie Mellon University Software Engineering Institute. The SEI is a federally funded research and development center sponsored by the U.S. Department of Defense. A transcript of today's podcast is posted on the SEI website at sei.cmu.edu/podcasts.

Suzanne Miller: Welcome to the SEI Podcast Series. My name is Suzanne Miller, and I am a principal investigator in the SEI's Software Solutions Division. Today, I am joined by my friends and colleagues Jerome Hugues and Joe Yankel. Jerome is a senior architecture researcher, and Joe is a senior engineer, both of whom work with me in the SEI Software Solutions Division.

Today, we are here to talk about their work with ModDevOps, which is an extension of DevSecOps that embraces model-based systems engineering (MBSE) practices and technology. I am very excited about this because I love the conjunction of these two concepts. I want to welcome both of you. Let's get started by talking about, why are you here? What brought you to the SEI? Tell us a little bit about yourself and the work that you do here, so we can get other people excited about coming to the SEI. Joe, why don't I start with you on that one.

Joe Yankel: Thanks for the intro, Suzy. I came here to be part of this ecosystem. Really learn, expand upon what I learned in industry with software development and testing and help apply that to our DoD partners.

Suzanne: That I think is the unifying thing for many of us: DoD, DHS, those sort of government partnerships that we have. Jerome, what about yourself? Where did you come from? And you are not from Pittsburgh are you?

Jerome Hugues: No, I'm not. I'm not originally from Pittsburgh. I am actually from France as you can hear. Well, actually, before joining the SEI, I was a professor of software, computer science in general with an emphasis on model-based and safety-critical systems. Fifteen years ago I started working on this AADL technology, and I became an expert in Europe. I got the opportunity to move to the SEI, not Pittsburgh actually. I am working from the Arlington office. Applying model-based techniques to the very challenging problems from industry on this side of the Atlantic was a very great challenge. I was really impressed by the type of challenges that arise on a daily basis. I started my work for the SEI working on model-based, working on AADL, and working on making other ideas that we have on model-based transitioning to the industry and customers that the SEI is supporting.

Suzanne: For those that don't already know, AADL is Architecture Analysis and Design Language, which is one of the implementation strategies for model-based engineering that the SEI is very involved with, just to clarify that for people.

Mod in ModDevOps stands for model, right? Let's talk about how does adding model-based engineering change the way we think about DevSecOps? What is it about model-based engineering that we are adding into the DevSecOps pipeline? Jerome, why don't we start with you on this one?

Jerome: Actually it is going in both ways. From my perspective, I am extending MBSE, model-based systems engineering with DevSecOps. On the other hand, we can also do quite the opposite, extending DevSecOps with model-based. It's really a question of perspective and play on words if you will. From my perspective, coming from the MSBE community, coming from SysML, AADL, and others, when I do modeling activities, when I create my system, I start from the abstract. I go down, refine my models and at some point, I will start implementing it. As I am doing that, I accrue benefits from early analysis that we can do on models. But also at some point, I want to go to the implementation. This is where the connection with DevOps or DevSecOps is relevant because DevSecOps will allow me to go further into developing my system, implementing it, and getting back metrics on my system so that eventually I can reflect on my design. So that is really one way of combining things, extending MBSE with DevSecOps. Joe probably can give the opposite extension, which is you are the expert on DevSecOps and all the benefits from models.

Joe: Certainly, I think you said it correctly. We understand these best practices that DevSecOps gives us. But what it really lacks is a formal method to understand the needs of the ecosystem. We know we can gather data well. We can monitor that data. We have metrics. But to really capture those things, you have to formalize it via a model. The model can go much deeper, it can help an organization not only understand requirements, capabilities, but even the organizational roles that need to be in place to do these activities. It's a very powerful tool.

Suzanne: One of the things that Fred Brooks, many of us know him from *The Mythical Man-Month*, but he is also the author of a book called *Design of Design*. I am paraphrasing, but one of his statements is along the lines of, *You can't really understand a design until you start to implement it*. Both of you are talking about that from different perspectives. *I can't understand my model until I start seeing what happens to it when I actually implement pieces of it through a DevSecOps pipeline*. And, Joe, *I can't really understand my design until I have a model that I can use as a reference point for understanding if what I am measuring and monitoring actually has any value or meaning*. So, this is an evolution of this idea that we need to implement to understand design. But we have to do enough design, so that we actually can implement something that is meaningful. That is what has got me excited about this extension of these two concepts or the conjunction of the two concepts as we go forward.

Jerome: Suzy, to elaborate on that, there is something relevant.You mentioned *The Mythical Man-Month*. Everybody is concerned about how many man-months does it take to implement a system. In both MBSE and DevSecOps, we advocate a lot for automation. With a model, we can do early analysis. The model is not perfect, but you can already remove a collection of risks. This is what the SEI has been doing for instance with AADL that you mentioned, and ACVIP, the virtual integration process. When we have a model, we can also perform some code generation. We can do a lot, I mean everyone knows SCADE, Simulink, even AADL code-generation practices. When we start doing that, we can go further and say that the code that we are generating can be part of the DevSecOps process that will kick in after that, so that we can go further and more importantly faster into the regular system. So this aspect of automation means that of course we still need actual, human months, but there are a lot of human months that actually get shrinked thanks to automation.

Suzanne: And we turned the mythical man-month into the mythical man-hour.

Jerome: Oh, yes.

Suzanne: That is a goal for a lot of people, that fast information. One of the things I am seeing as we have more connection between software and physical systems, what we call the cyber-physical systems [CPS], when you make decisions in hardware, they are more difficult to change later on down the road than the software. To the extent that we can do better models of our hardware and that aspect of hardware design in these cyber-physical systems, we get better data to make those hardware decisions at a time when we still can afford to change our minds. In some of the systems I work with, that is actually very important. That is one of the most important things about modeling is that we get information that we need so that we can make better hardware decisions before they are too late to change. OK, so I have actually answered one of the questions that I have for you, which is, what is some of the value of working in a digital engineering environment that applies MBSE, and what kinds of things, besides what I've already talked about, does it make possible that can't be accomplished in more conventional environments? Joe, I'll start with you this time.

Joe: All right, I will take a crack at this one. What I see happening today is we are playing that game of Whac-A-Mole. We are seeing a problem. We are responding to it. We often throw a tool at it. Modeling can really help us here. It formalizes these planning activities that have to occur. We talk about the actual requirements of the system from all the stakeholders. DevSecOps says, *We are going to collaborate with everybody*, *so I really need to know on the hardware what kind of things it does*. If we are talking about measurement, *Well, I might affect how we measure that hardware*. We have to have a system capable of capturing those things. So we are not only concerned with the application or system, we are also concerned with the system that enables the development of the system, so the system of systems so to speak, multiple pipelines. That is very complex if you don't formalize your approach. And a model is about this; it allows us to look at all the requirements. We have a good idea of what DevSecOps is, but we need a source of truth. A model can provide the source of truth that, *Here is all the goodness. Here is how we are addressing security. Here is how we are addressing what may be the core pieces of DevSecOps*. We need to do those things, and we need to quit doing them in silos. So, even as we have recognized now there are a lot of great practices, how do we combine those and build upon them? We put them in a model. It will enable the future resilience that we really demand.

Suzanne: Yes, and I am hearing you talk a little bit about quality attributes, which the SEI talks a lot about. The quality attributes are elements that go across different silos of a cyber-physical or just a cyber system. Security, and usability, sustainability, all those things that make a difference in the overall product but can't be captured in any single element of the product, and models allow us to see the effect of changing a security attribute. *If we loosen this up, what happens?* And models allow us to look at that, and DevSecOps allows us to test it and say *OK, let's try that*, *let's see what happens with the code if we do that*, and monitor and run our simulations and all rest of it. So that aspect of modeling is I think something that a lot of software engineers may not think about in terms of how it enables them to actually achieve some of the quality attributes that they sometimes kind of throw their hands up and go, *Well I can't implement it in this one component, but it affects all the components*. What else would you want to say about working in digital environments and how it makes it easier for the systems and software engineers to accomplish their work, Jerome?

Jerome: Well, before that perhaps, Suzy, we can just finish on ModDevOps. Really as Joe and I and others on the team, we wanted to define it really as an extension of DevOps and DevSecOps.

That is why we change how we act, if you will. The DevOps definition as provided by the Air Force so that we can just change a few words saying that really *ModDevOps is a systems–software co-engineering culture.* And it is really this future aspect that Joe mentioned before. We have a collection of stakeholders. They have to collaborate and work together to achieve a common goal. For that we use model as a support to some DevSecOps activities. As we were working on this project during FY20, what we did is not only models of the system, but also models of the pipeline that we go from model to the actual system. It is really this aspect that can help a lot of digital engineering activities. Think about this system you want to produce, but also all you want to produce it, all you want to collect, all the quality attributes. Some of them will be evaluated on models. Some others will be validated on the implementation. Some others will be validated on the final deployed system. I think a clear picture of where you get [this] information, all that can be combined so that you can convince the customers that, *Yes, I have done my duties. I am developing the right system, and I can demonstrate why the system will work as expected*. By collecting also that data across a complete system lifecycle that with engineering, and production, and deployment is really the greatest value we can get, which is really to understand what is a system, how we build a system, and why we believe the system is correct.

Suzanne: You are basically describing enlarging the ecosystem. Joe talked about this earlier, that there is a whole ecosystem around this, and you are enlarging that ecosystem. So my challenge to you is to take on the next evolution of that ecosystem into digital acquisition. Because right now we are really talking about the engineering part of a system. But around that is all the things that go into, *How do we decide what the right system is?* And, *How do we get the right people to build that system?* And, *How do we get all those stakeholders to agree on what the quality attributes are?* So that is for the next six months or so. You should be able to just knock that out. I am teasing a little bit because that is actually the next step. We are seeing programs that are trying to embrace digital acquisition, but my assertion would be you have got to deal with the ModDevOps from an engineering viewpoint first so you understand that ecosystem before you can really understand what the next level of that ecosystem is.

So anyway, all right. Let's go on to…We have been really up in the clouds with this in terms of ecosystems and things. Let's talk a little bit more down in the weeds about what would be the typical steps in a ModDevOps development process. What steps in that process would be different from what you would expect in a typical engineering process that results in a piece of software? Who wants to go first on that?

Jerome: Well, let me start with that. Surprising I would say that we won't change a lot of the steps themselves. Simply what we change is what we do during the steps. Basically, of course, we have to plan: *What do we want to do? What are the requirements? What are the properties of* *the quality attributes that we want to collect? What are the parts of the system? How do we want to engineer them? How do we integrate them, so that we can deploy the system?* So it is basically typical DevSecOps steps that you would want to perform. What would be different is, we need to consider at each step what benefit you can get from models. So, of course, when you start very early requirements, well, you can do requirements with SysML, you can do requirements using DOORS, you can do requirements on a chalkboard if you will. But you are starting [with] populating a database with quality attributes that you will collect and connect to the other steps. When you start thinking about the architecture of your system, when you start thinking about the quality attributes, when you do this three-month project that supported this activity, we took the example of a cyber-physical system. So we started with the big picture, SysML. *What is my system about?* And after that we refine it. We want to build an actual software out of our systems. Obviously, we would need AADL for that because it does a really good job of semantics for that. And after that you get example with connection to physics. We have two options. Particularly we could prototype it on Raspberry Pi, which is great. But we would have needed thousands of them for complete large-scale system. On the opposite what we decided is to use Modelica, which is also a modeling notation, in this case for physics. After that when you have all those models, the first question is, *Why? How do you interoperate? How do you integrate all of that?* And again, through model-based techniques, we can make them interoperate. We can generate code from AADL, we can generate code from Modelica, and we can integrate to build digital twins. And we have demonstrated the feasibility of that through this virtual-integration approach that we are through model-based techniques and code generation from models. And after that we apply the other steps of DevSecOps which is, *OK, we want to have CI/CD [continuous integration/continuous deployment] techniques. We want to automate the deployment. We want to automate the packaging. And we want to collect data*. So it is basically DevSecOps with a little bit of models. But the models that we built have been selected because of the problems that we wanted to resolve, which was the CPS, and therefore AADL and Modelica were a perfect fit for that.

Suzanne: Did you want to add anything to that, Joe?

Joe: I think Jerome said it great. Some of these DevSecOps processes helped enable the repeatability of this. Core concepts like infrastructure as code, configuration of code. And what we talked about. We want to do this early. We want to prototype. So this is really all about shifting left, failing fast. We don't have to have the model perfect. But we have to begin doing these things to improve it continuously. So in our minds, we can have a model that goes through iterations. We want to make sure our system can support those types of improvements. If we do it early enough, we have a lot of confidence we can do this later, so sustainment, maintenance. We are addressing those things now, not in the future.

Jerome: Another topic you mentioned a lot of DevSecOps concentrating of something *as* code, infrastructure as code. But if you take it to the model, because we can generate code from model, and this code can be integrated, to some extent you can have model as code. To me model is nothing but I would say a more abstract notion of source code. To the point where you can say for some classes of systems, you can have a system as code, which is a little bit I would say shocking for some systems engineers to think about it. But for the class of CPS that we are working on for IIoT systems, for many, I would say even IT systems, there is no real difference between the system itself and the model we are presenting here. So, thinking of system as code is really one of the results that we have demonstrated as something tangible through this TwinOps project. It is really a proof of concept demonstrating that, *Yes, it is achievable*. Now the key question in terms of transitioning that is really *OK, it worked on a small-scale example. How could it scale at the enterprise level for the DoD?* That is a gigantic challenge ahead of us, but from what I have seen, I am definitely confident that it is something that is achievable through digital engineering and digital thread initiatives.

Suzanne: One of the things that comes up in that for me is, many of our viewers may know that the DoD did initiate a digital engineering initiative in the NDAA [National Defense Authorization Act]. I think it was 2018, I am not exactly sure. So if you're in the DoD, there is leadership support for moving towards digital engineering. One thing that we haven't talked about is the difference between applying this in a new system being built versus applying it to legacy systems. One of the challenges that we know about legacy systems is creating models is essentially a reengineering process of understanding what is in the current system, so that you actually can start to interrogate it and start to ask questions about it. Is there work going on in that aspect? Because the project that you guys worked on was more of a new set of requirements. It wasn't a legacy system that was being built. Can you speak to that for our viewers that are in the legacy environment, so they have an idea of how this might apply to them or might not?

Jerome: I can just say a couple of words on that. So if we take aside ModDevOps and if we look at what we have done with AADL, in the past some members from our team, the model-based team, took existing systems. They did a bit of reverse engineering to demonstrate specific attributes. Like for instance, the capacity to change the CPU architecture without impacting the scheduling of the real-time properties of the system. So the question really depends on the type of questions you have. Of course, the more precise the questions, the more work you need to do with these reverse-engineering activities. But from a model-based perspective, really the question is to recognize which questions can be answered through this reverse-engineering effort. You are right, at some point the cost of modeling is so high that redeveloping the system is the only option. The same way that when you take legacy software, there is a tradeoff between recompiling it for a new CPU platform versus redeveloping it. We are exactly in the same situation. It is really a tradeoff, and we have to collect metrics to know what is a decision point. We are not really at this stage yet in the MBSE community I would say.

Suzanne: OK that is fair. Joe, anything to add on that topic?

Joe: Well, you hit on an important topic. It is harder with a legacy system. There are a lot of considerations. I do believe since we have done this project, we have also looked at just modeling what DevSecOps is just by itself. Like, what are these core attributes that we define them as a model? Now, this approach, this would allow you to evaluate a legacy system, to understand what capabilities it may lack so that you could use modeling to help pinpoint places of prioritization, places where, *Ah, this is a great practice, we're not doing it. Let's focus on improving it here.* It can be done. It's going to have to be done with legacy. Certainly, some things are easier. Some architectures are simpler to do this with too. So complex systems of systems is a challenge. We do know this, if you started with model-based it will be easier to proceed and update. Retroactive fitting of this is a challenge, and it has to be evaluated on a per-system basis.

Suzanne: Sure. One other question. You brought up sustainment. One of the things I know is an issue in model-based engineering in general, and I am curious as to how you see it when you bring DevSecOps into it, is the issue of keeping the models up to date. From what you said so far, my impression is that this is actually easier if I have modeling as part of the DevOps pipeline, because I am going to have some feedback loops that lead me directly back to the models themselves. Can you comment on that in terms of how ModDevOps as a strategy improves your ability to keep the models up to date and relevant?

Jerome: The interesting aspect if you apply this idea of a pipeline that we integrate from DevSecOps is also that with this pipeline is coming, I would say, a list of responsibilities. The who-is-who of the project. *I am in charge of the CI/CD. I am in charge of defining requirements. I am in charge of defining the metrics.* And you can apply this also in the models. *I am in charge of defining the architecture.* Beyond those responsibilities there is a collection of models. So from the metrics you can identify which portion of this pipeline has to be updated. Do you have to update something in the software model, on the architecture model? It will be a discussion between those experts. Definitely the benefit is that because we also capture the process, we have captured the complete pipeline, we know exactly who is in charge of what, so that we can do an impact analysis, and so that we can put around the table the right set of people saying, *OK we have this issue. Who is in the best position to address it? Are you the software guys' architecture?* They can have this discussion and know precisely how to react, I would say.

Suzanne: Joe, anything to add to that?

Joe: I would say he nailed it there. With a model, we have operational process flows. These include the planning sessions. Part of planning is saying, *How are we going to look at the data we've decided matters to us*? You plan the metrics. You plan the data that could influence the model. Some of this data as part of these sessions will be, *We are not performing well*. Or, *Something is happening, we need to update the model based off this information.* Continuous feedback is useless if you don't have continuous evaluation of that data. That data can potentially affect the system itself, or it could affect the system that builds the system or that enables the development of that system. If anything, the model could help enforce, *This is how we operate. These are the process flows that continue through the end of time for this system.*

Suzanne: We are getting the continuous evolution. We have got continuous integration, continuous deployment. Now we are moving into continuous evolution. We have got I, D, and E. So there you go.

Jerome: Actually as a continuation to this idea that you push, Joe, which is to remember that we have this feedback loop. But as we are doing models, in addition to implementation work, we can actually have as many subset [feedback] loops as we want. For instance, I can have a subset group between SysML and AADL. SysML will give me the overall architecture, and I can propose and evaluate multiple designs. We can reject one because it is too costly. Because, well, if you don't have the board it can make a big difference. Or the energy consumption may not be the right one, or the interfaces may not be the right one. We can have this micro loop. We can add another one for instance that we experimented a little bit between AADL and Modelica for representing the sensor or what type of metrics we want to collect from the environment: the temperate, the pressure, etc. This idea of CI/CD is not something…. We should not reproduce a waterfall model between the top of the system and the final system. We can have as many intermediate feedback loops as we want, since the automation. It can happen any time we have two models that are interpreting information, or even using one model. That is something that is also striking to think that we can automate and define as many intermediate pipelines as we want. That is also a great added value of this approach.

Suzanne: I am hearing two things. One, the pipeline is itself a model, and we can keep evolving that model as we get data about the pipeline. I am also hearing the model is a pipeline. I am hearing both sides of it. The model shows me the pipeline, helps me to make decisions as we go along. What we are really talking about here is a different way of doing decision-making in engineering. I am hoping that our viewers are just as excited about this as I am. The thing I want to close our discussion with is, if I am somebody who is excited about what I have heard here today, and I'm working on cyber-physical systems, any complex software development, where do I find out more about this? Where should I start? What resources are available to me through the SEI or other sources so that I can actually get started doing something with this? And I'll start with Joe this time.

Joe: All right. So the SEI does publish some resources. Through those papers and the publications we have you can reach back to the SEI to get additional information. Our current research, I don't believe that is all public. But we are still currently researching this. This is a difficult effort. It is ongoing, but we do expect to have some public releases of some of the DevSecOps modeling we are doing now. It is going to need your feedback. So we will appreciate anyone that is interested to give us that feedback so we can make it right. We also think this should be…. We want to fail fast here. We don't think it is going to be perfect, but we want to get there by putting it out there and trying it. That is one of our goals. Let's prototype. Let's build, Let's test. Let's get better.

Suzanne: What about some resources in the modeling, Jerome?

Jerome: In terms of modeling, we have a lot of resources in the digital library at the SEI on these topics. There is a technical report of some prework that we did. I would also tell you to watch this video. Generally speaking, ModDevOps as we introduced it is really a field of view much more than a collection of technologies. Let's discuss that. In my view it is probably the approach to exchange and see how it applies to your specific problem. The more problems we can discuss with you, the bigger we can refine all those concepts, because we really believe that there is a big push toward this direction, but it will take lot of time to have an enterprise-wide agreement across the computer industry. But we strongly believe that there is something very important to look at for the future. Yes, and in terms of activities, continuing applying some of those ideas and trying to improve the semantics of some models. But that would be the topic of a follow-up discussion, I guess, Suzy.

Suzanne: Absolutely. More discussions. I do love these kinds of discussions, and I want to thank both of you for being with me today and letting me take you far afield of probably where you thought you were going to go with this discussion, but I think it's so important. This is an important topic, and it really is part of the future that we are looking at. It is not just about artificial intelligence. It is about being intelligent about how we do the work, and modeling the work is one of the ways we get more information about doing that. And iterating, I mean just all the things come together for me. So you can tell I'm excited about this.

Thank you again for joining us. Thank you to our viewers. We will include links in the transcript to resources that were mentioned during this podcast, and there are a lot. If you start looking at model and DevSecOps, you will find a lot that are very compatible when you start looking at the work of the SEI, so I encourage you to go out and do that. Thank you again for joining us today.

Joe: Thank you.

Jerome: Thank you.

Thanks for joining us. This episode is available where you download podcasts, including SoundCloud, Stitcher, TuneIn Radio, Google Podcasts, and Apple Podcasts. It is also available on the SEI website at sei.cmu.edu/podcasts and the SEI's YouTube channel. This copyrighted work is made available through the Software Engineering Institute, a federally-funded research and development center sponsored by the U.S. Department of Defense. For more information about the SEI and this work, please visit www.sei.cmu.edu. As always, if you have any questions, please don't hesitate to email us at info@sei.cmu.edu. Thank you.